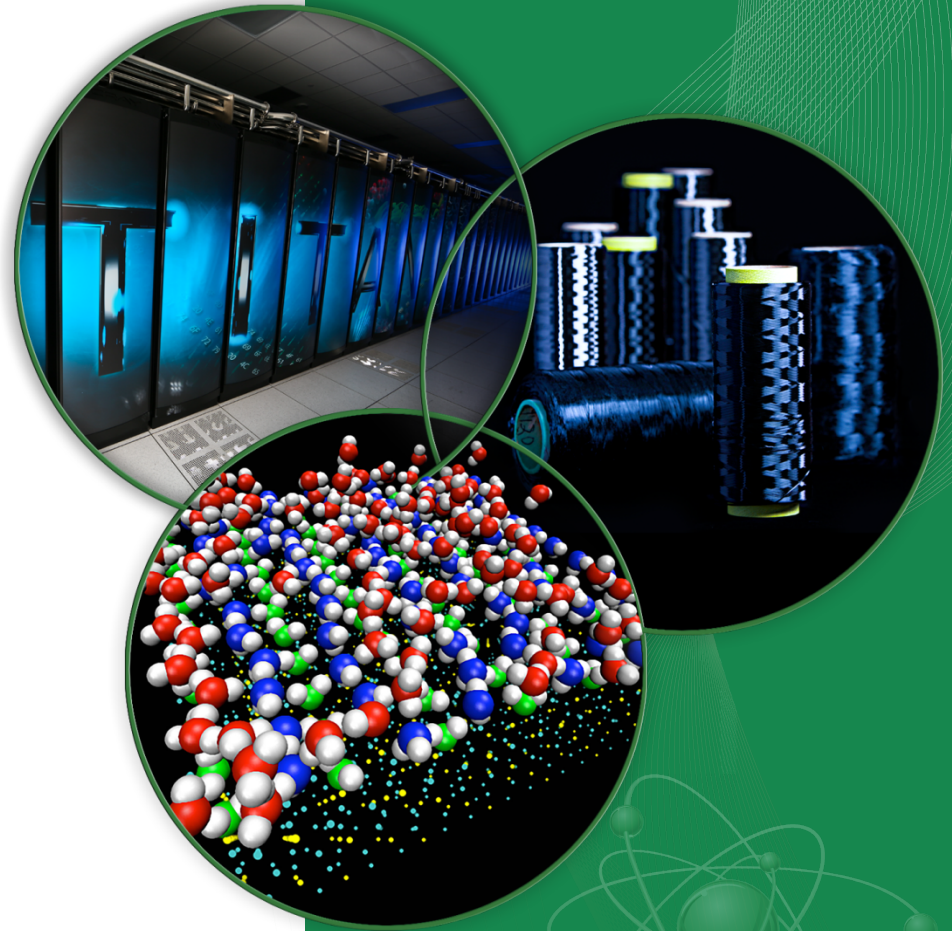


Lustre Networking Technologies: Ethernet vs. Infiniband



Blake Caldwell
OLCF/ORNL

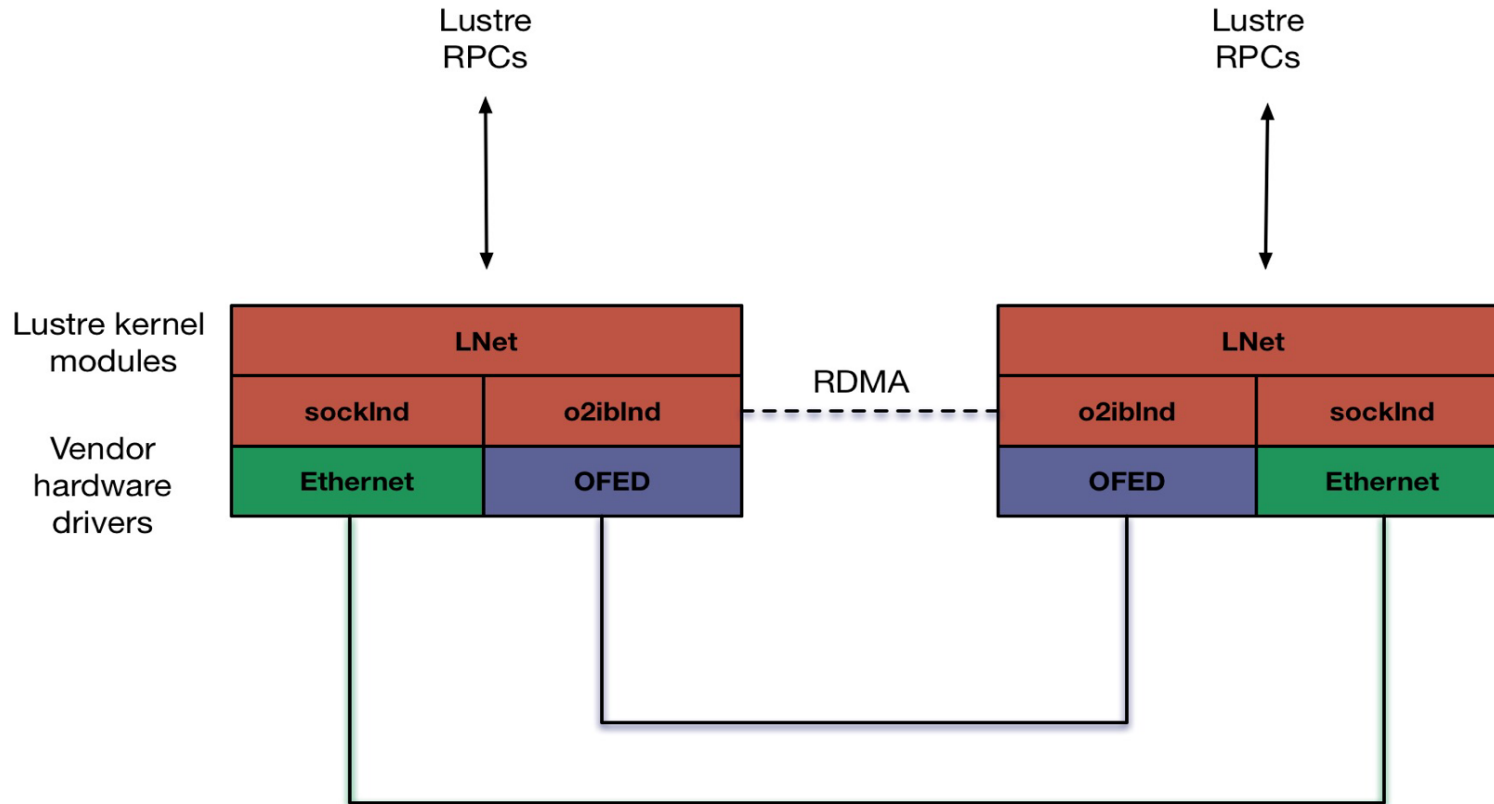
2nd International Workshop on The Lustre Ecosystem
Baltimore, MD
March 9, 2016

ORNL is managed by UT-Battelle
for the US Department of Energy

Overview

- **LNet Architecture Overview**
- **Comparing LND implementations**
 - **Infiniband vs. Ethernet (TCP)**
- **TCP LND Case Study**
 - **Results with 2x bonded 10GE**

LNet Architecture Overview



Infiniband vs. Ethernet Comparison

- Key L2 Differences
- Failure Resiliency
- Performance in Optimal Conditions
- Performance under Congestion
- Datacenter Network Integration
- Long-Haul Network Considerations
- Tuning Complexity

Infiniband vs. Ethernet Comparison

- Key L2 Differences
- Failure Resiliency
- Performance in Optimal Conditions
- **Performance under Congestion**
- Datacenter Network Integration
- Long-Haul Network Considerations
- Tuning Complexity

Key L2 Differences

Infiniband

- Guaranteed delivery
- Hardware-based **retransmission**
- Link-level flow control is credit-based
- Congestion control is native to IB spec
- Forwarding tables configured by SM before passing traffic

Ethernet

- Best effort delivery
- Hardware-based **error detection**
- Link-level flow control must be explicitly enabled
- Congestion control at higher level
- Spanning tree must converge (distributed algorithm)

Failure Resiliency

Infiniband

- No guaranteed delivery in the face of failure
- Failure will be detected by subnet manager
- Lustre supports active/passive bonding (failover only)

Ethernet

- Failure handling in transport layer
- Indirect failure detection through timeouts
- Kernel-level bonding
 - active/passive failover
 - active/active aggregation

Performance in Optimal Conditions

Infiniband

- Single active link in current Lustre releases
 - 55 Gbit/s (FDR)
 - 97 Gbit/s (EDR)
- Low latency to application through kernel bypass
- Fabric has higher bisectional bandwidth

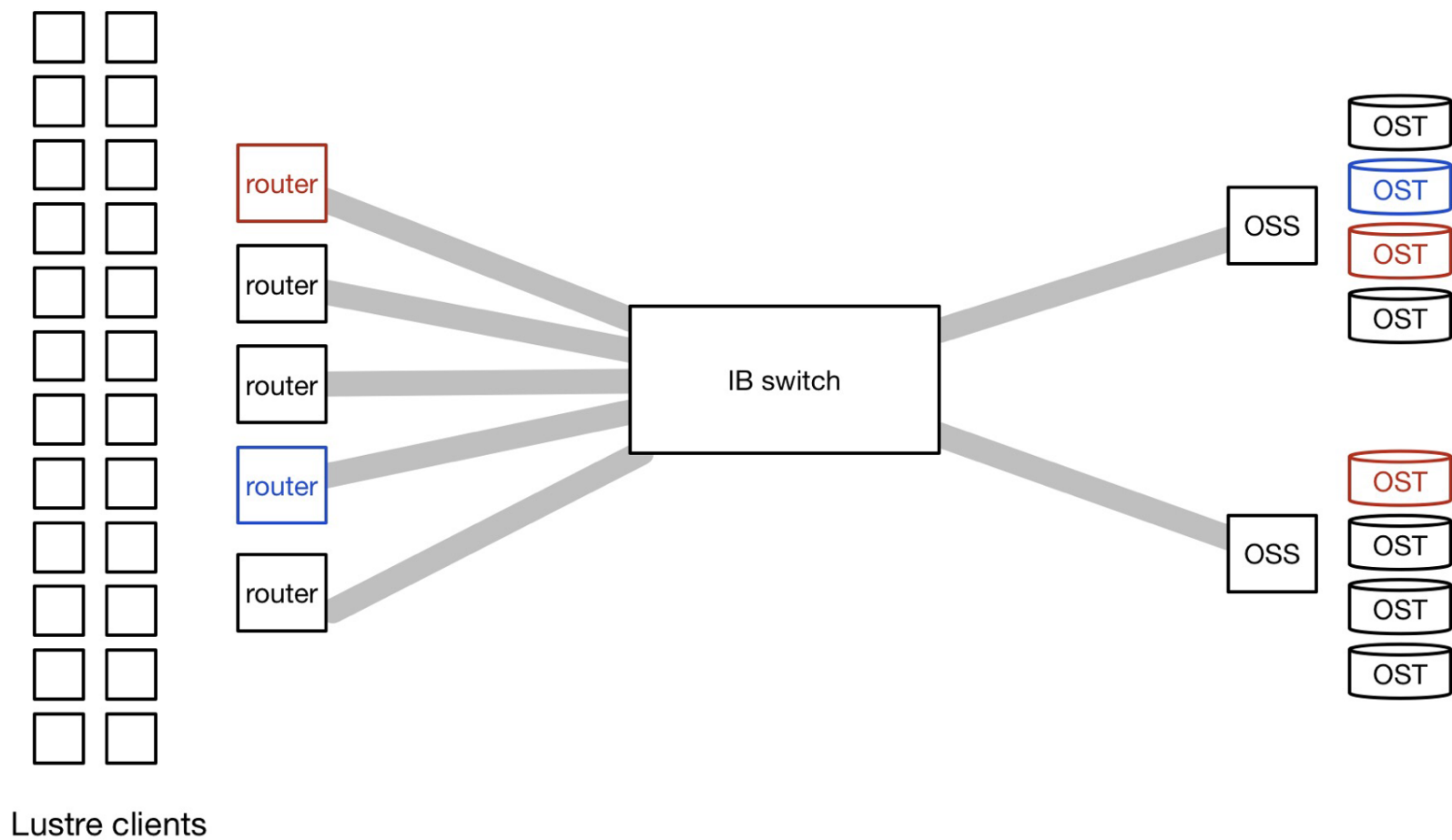
Ethernet

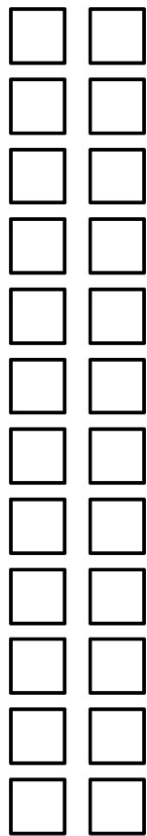
- LACP bonding native in Linux
 - 16 Gbit/s (2x10G)
 - 64 Gbit/s (2x40G)
- Context switches and buffer copies increase jitter
- Spanning tree leaves some links un-utilized

Performance under Congestion

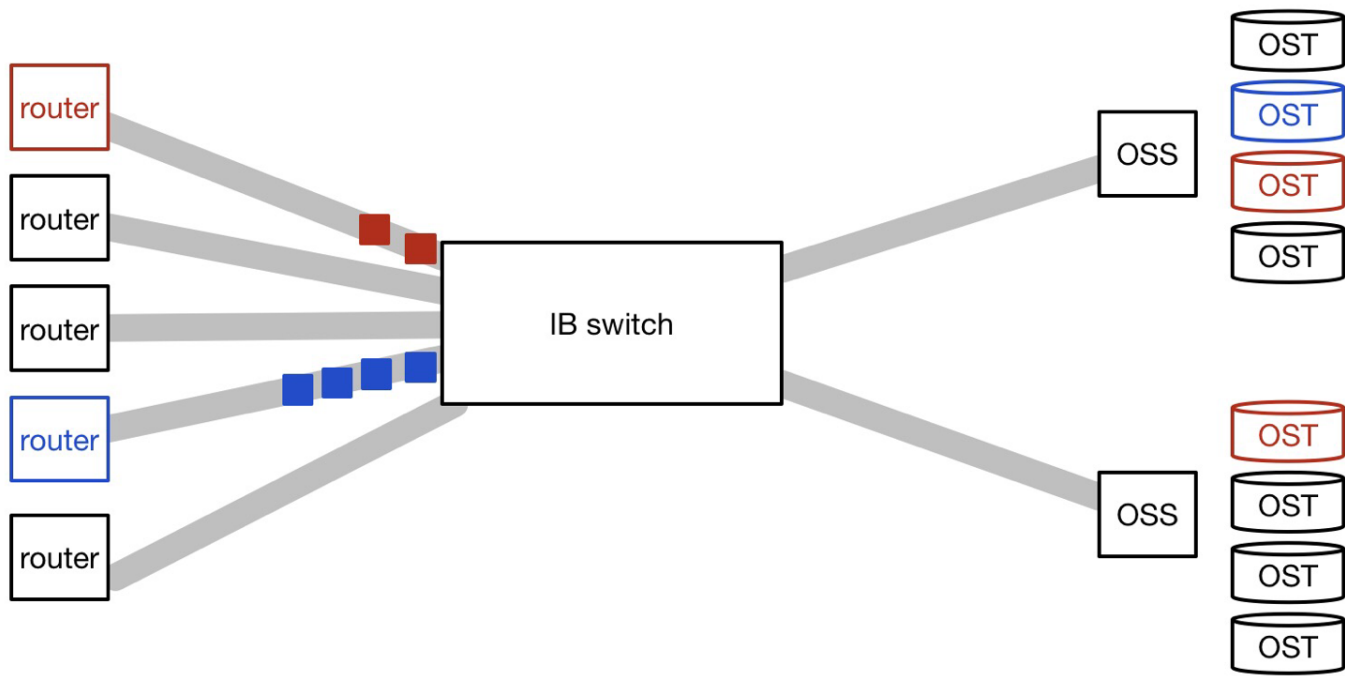
Part 1: LNET on IB

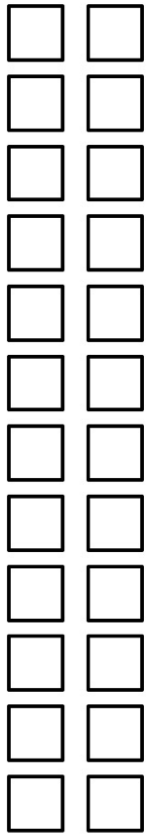




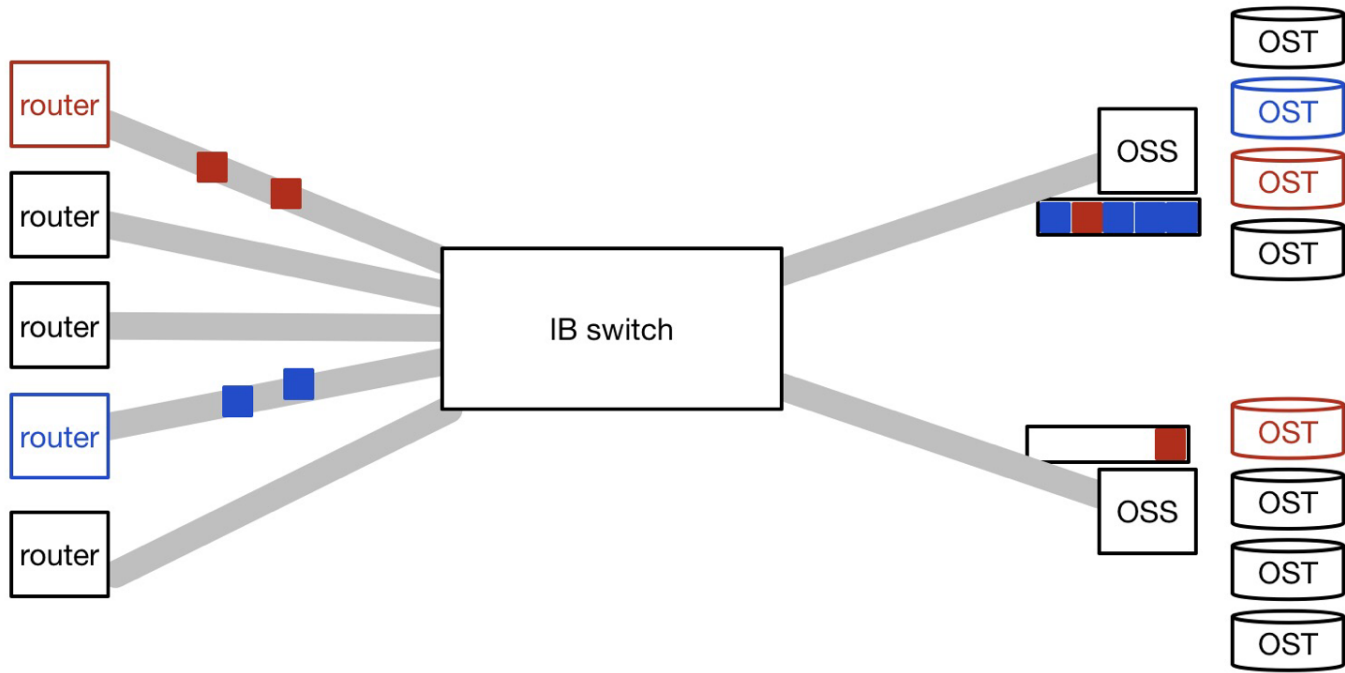


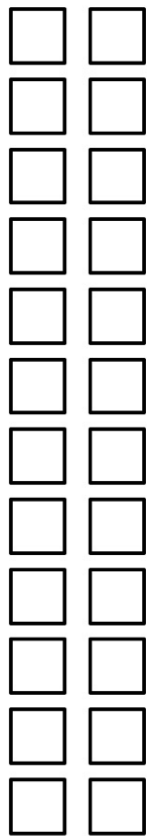
Lustre clients



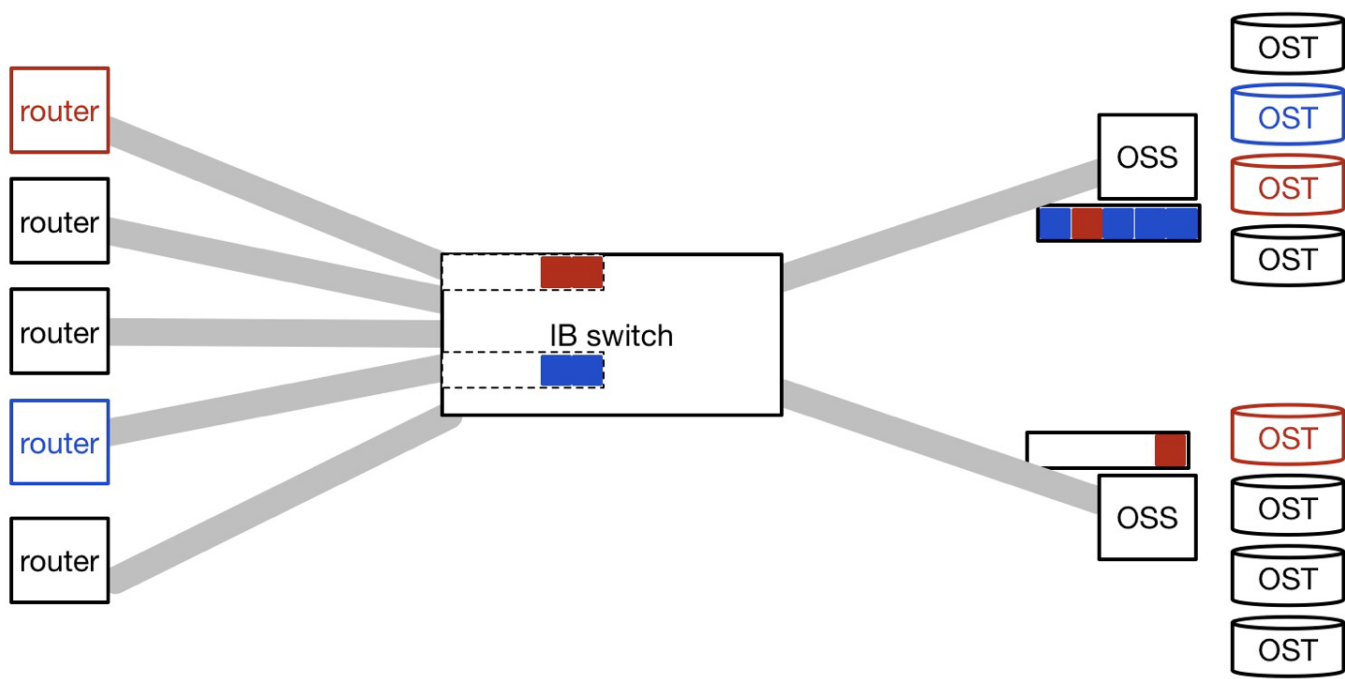


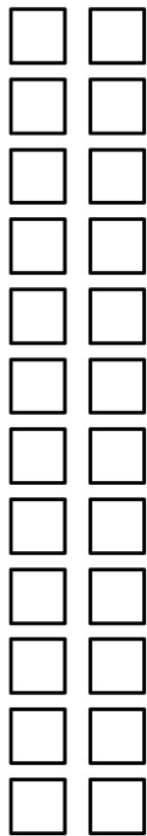
Lustre clients



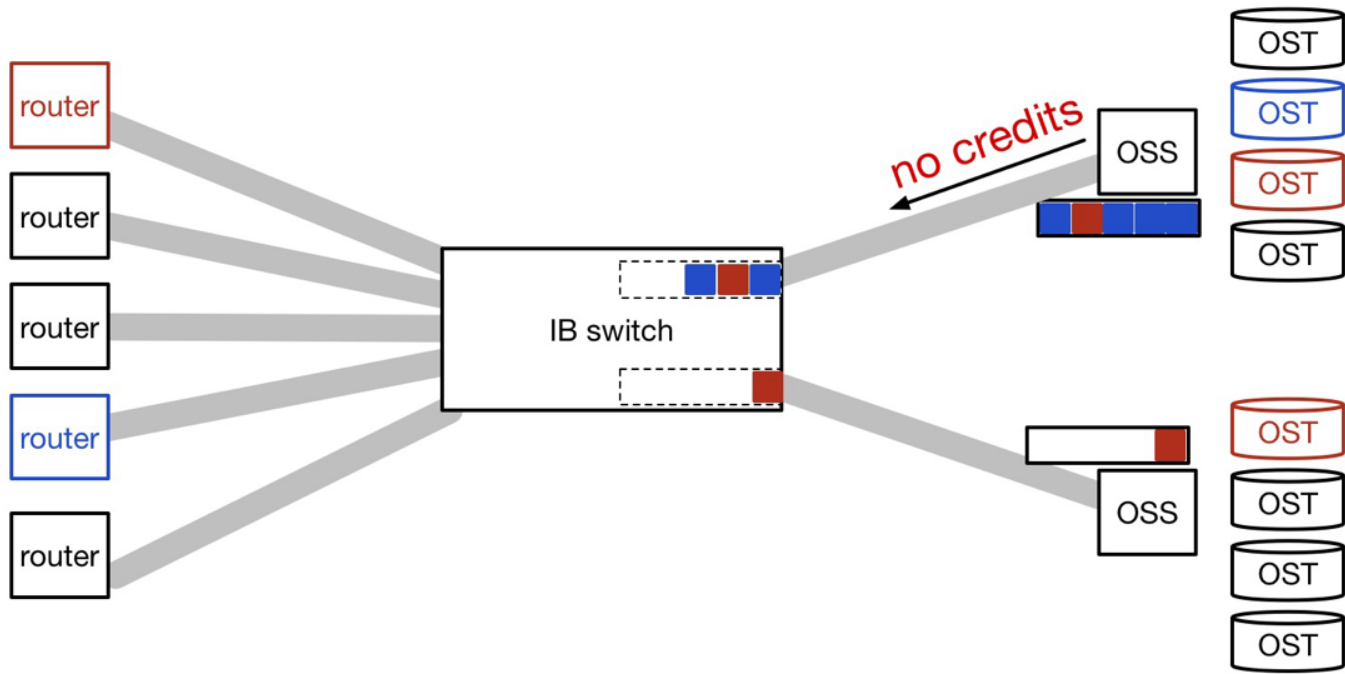


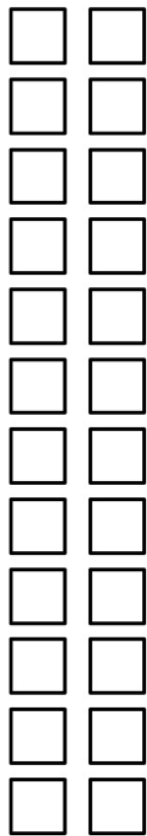
Lustre clients



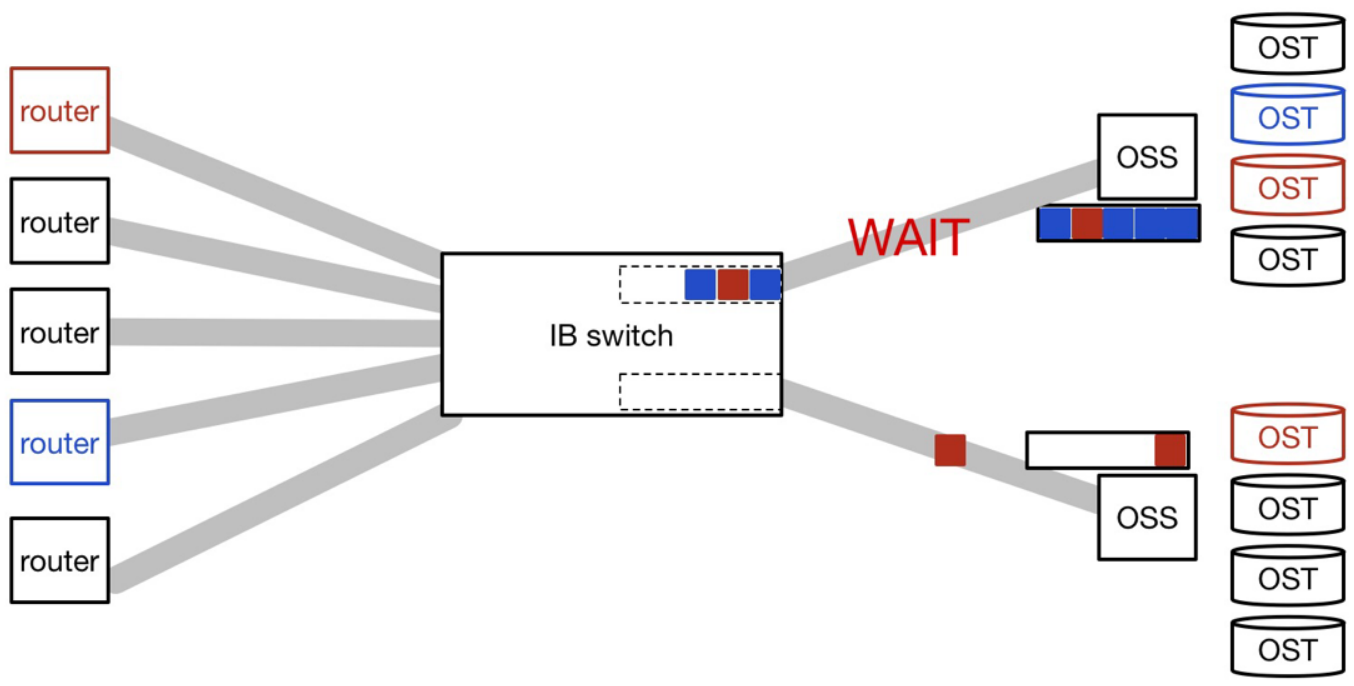


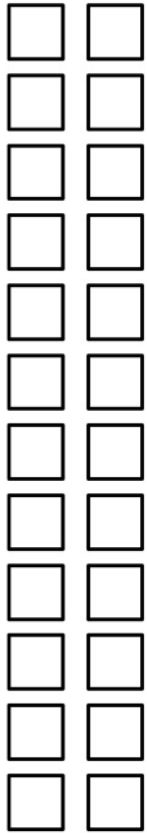
Lustre clients



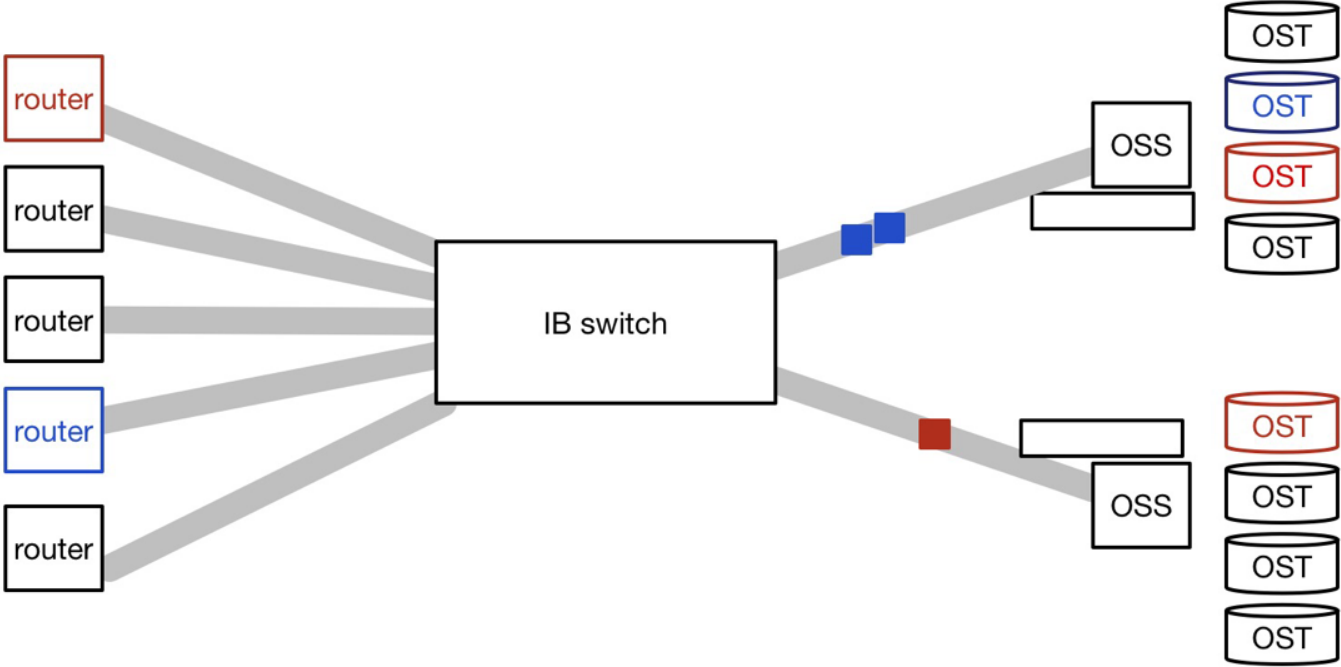


Lustre clients





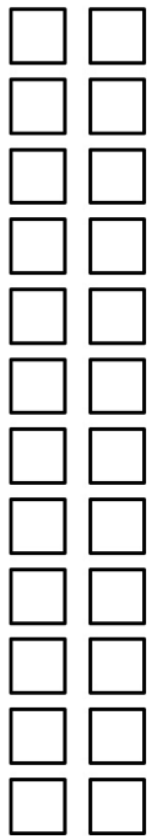
Lustre clients



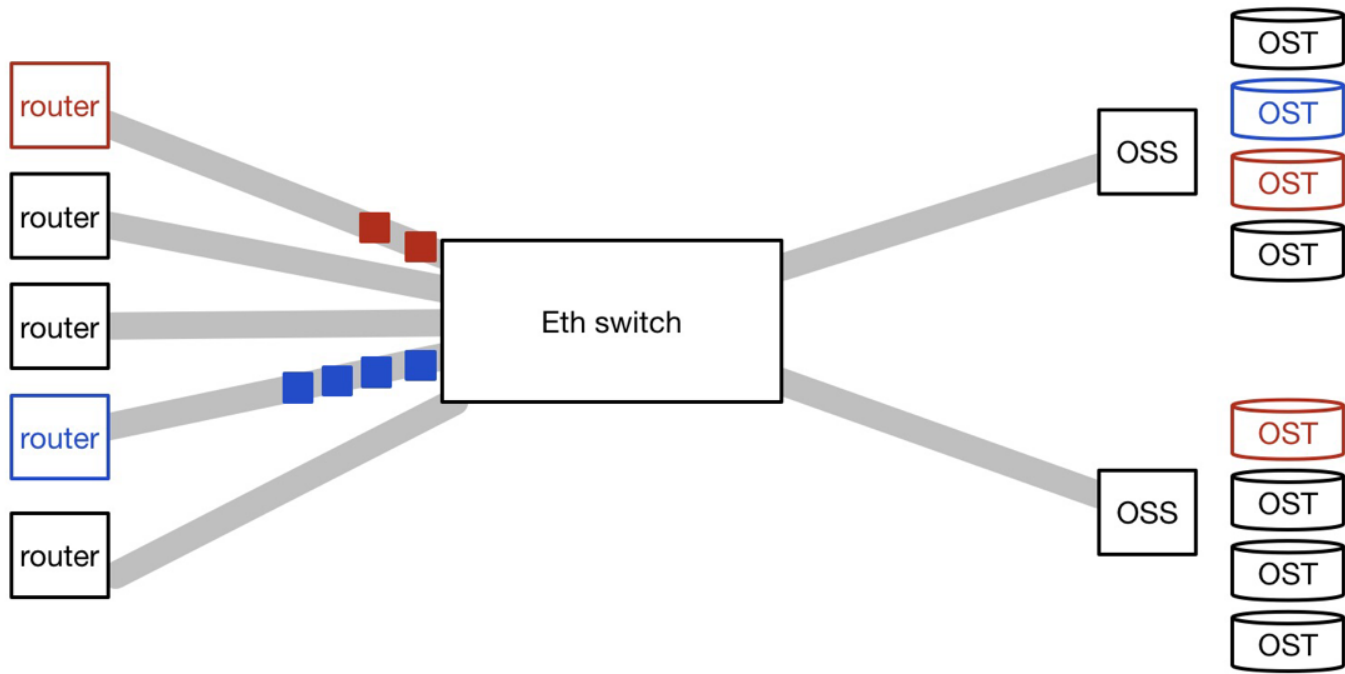
Performance under Congestion

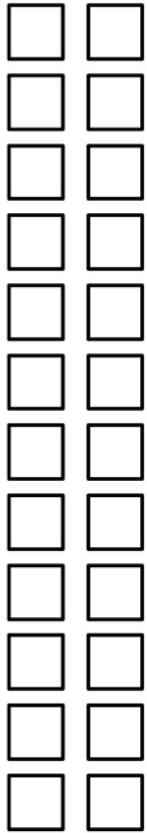
Part 2: LNET on TCP



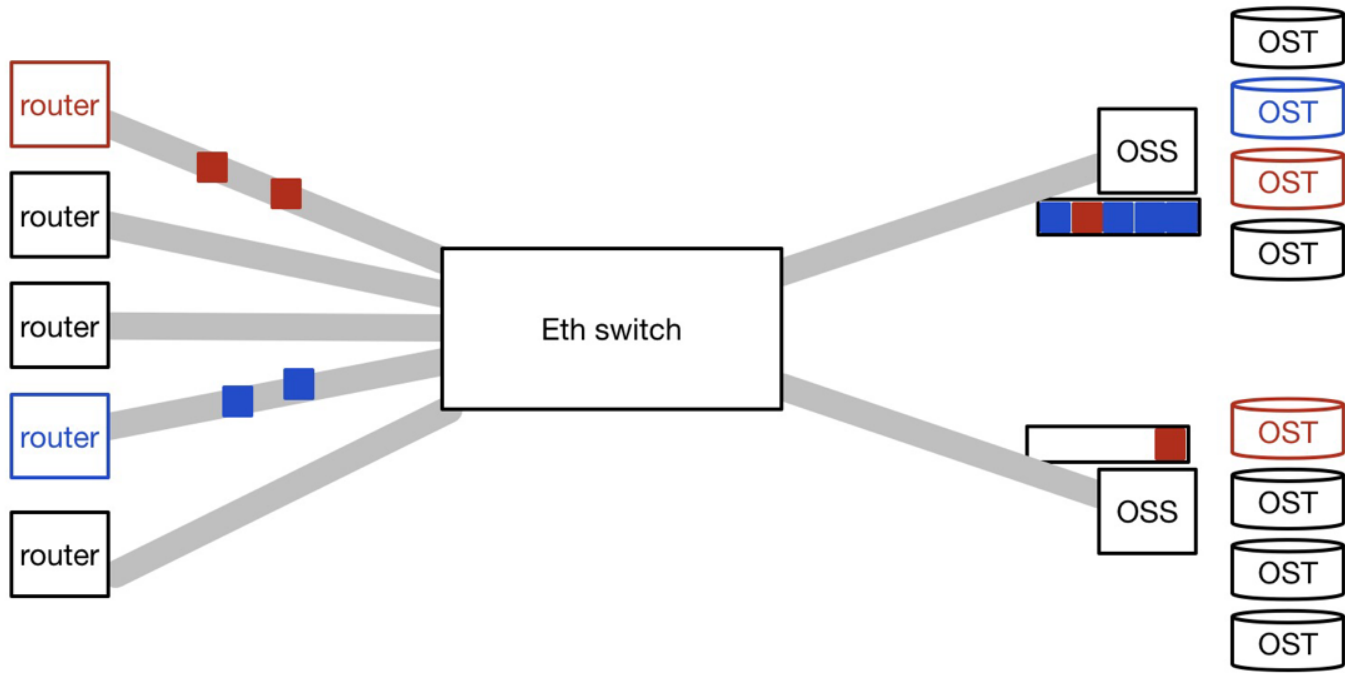


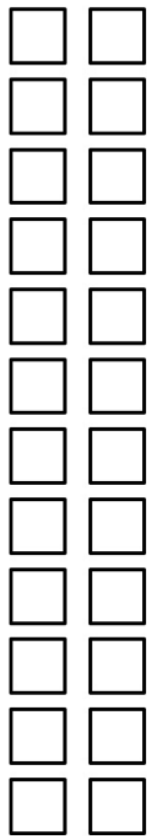
Lustre clients



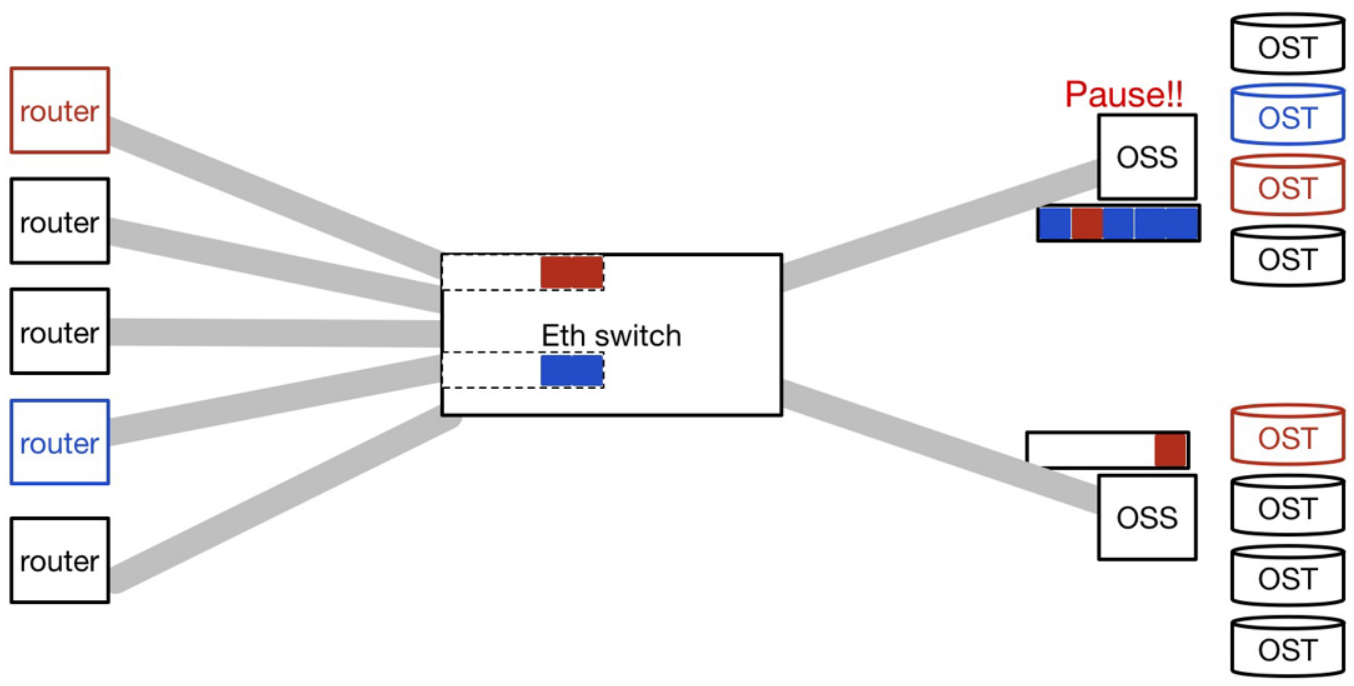


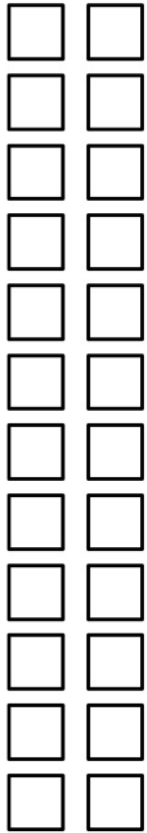
Lustre clients



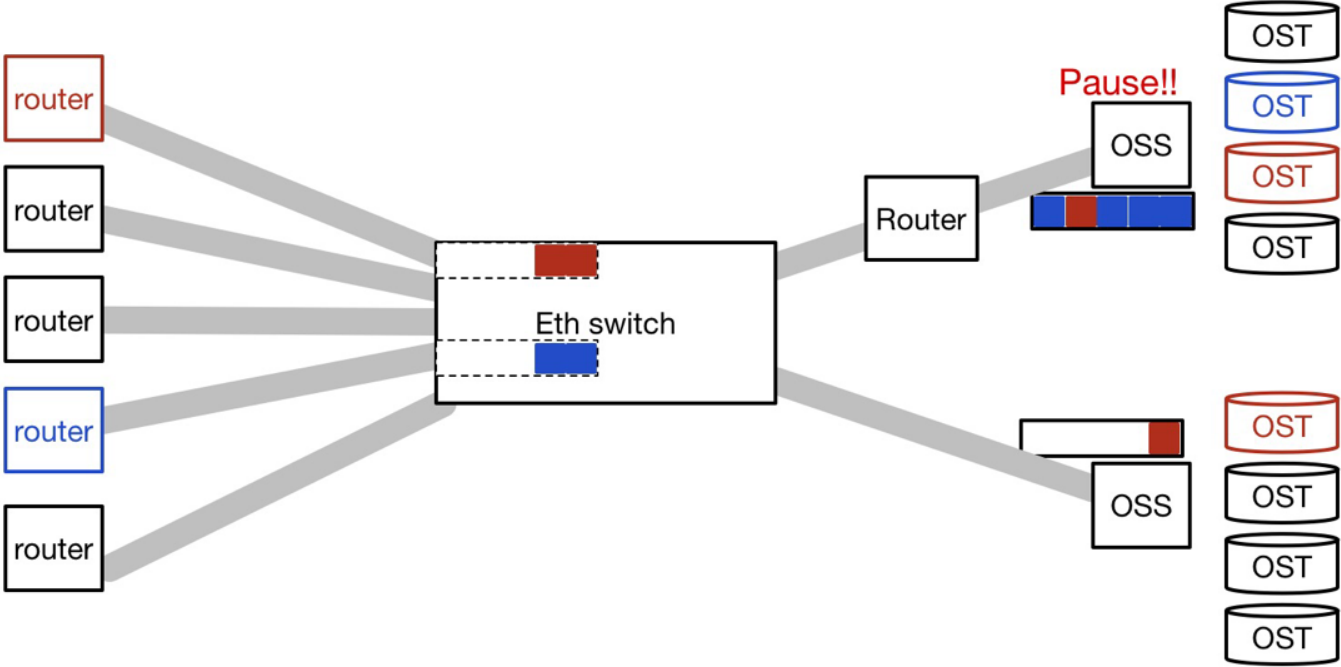


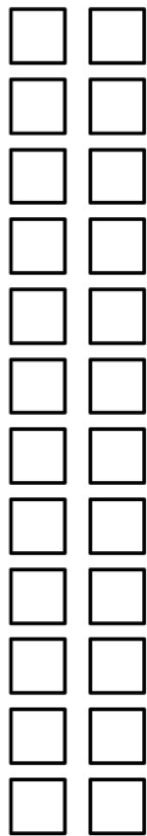
Lustre clients



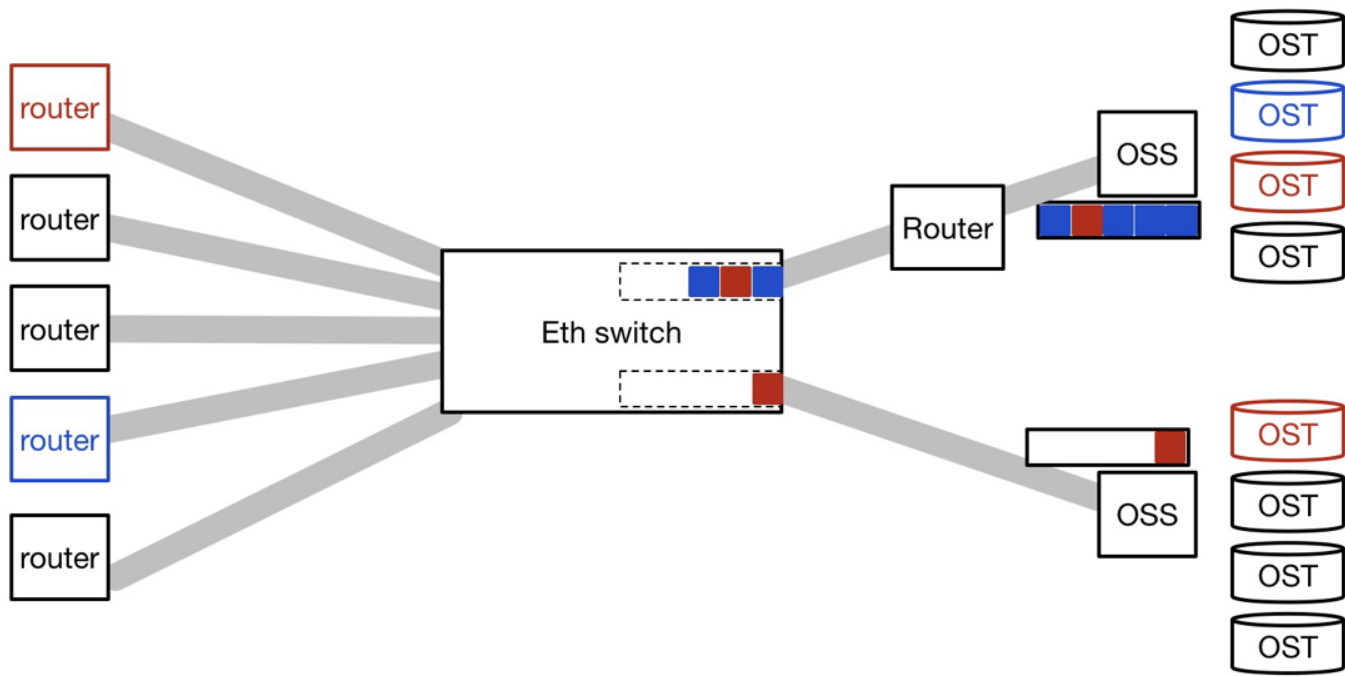


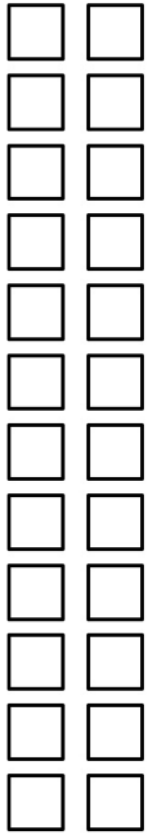
Lustre clients



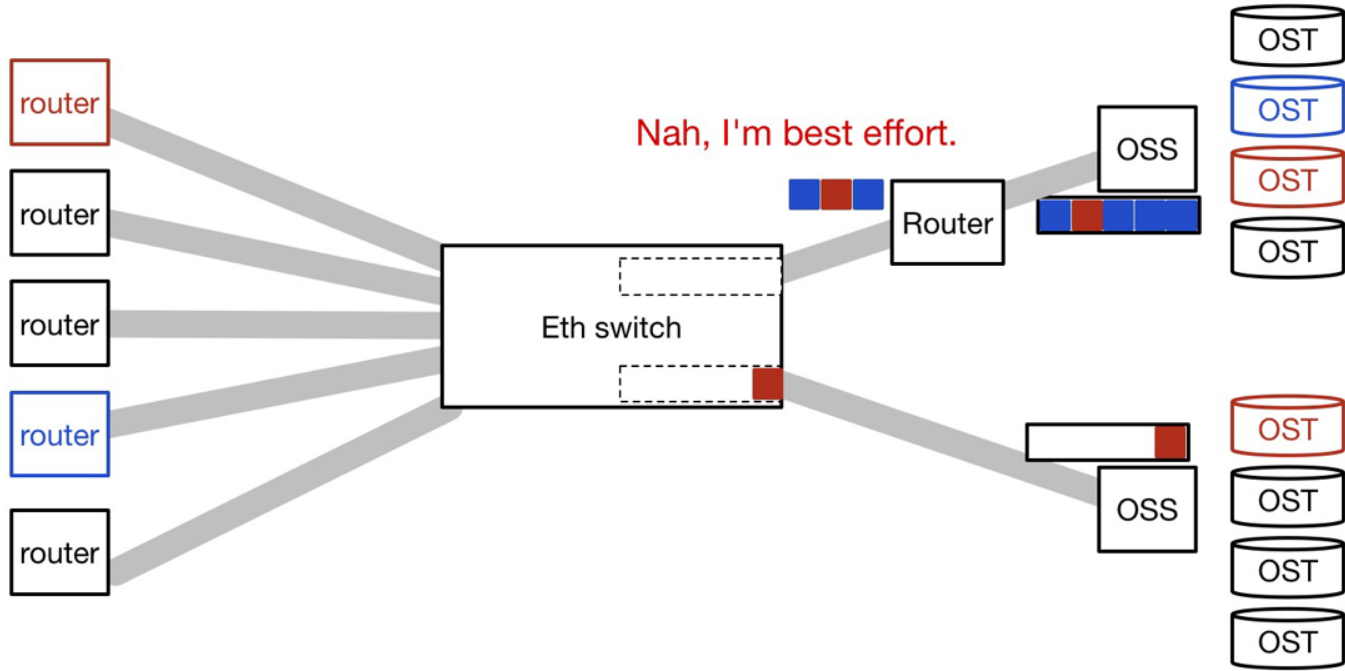


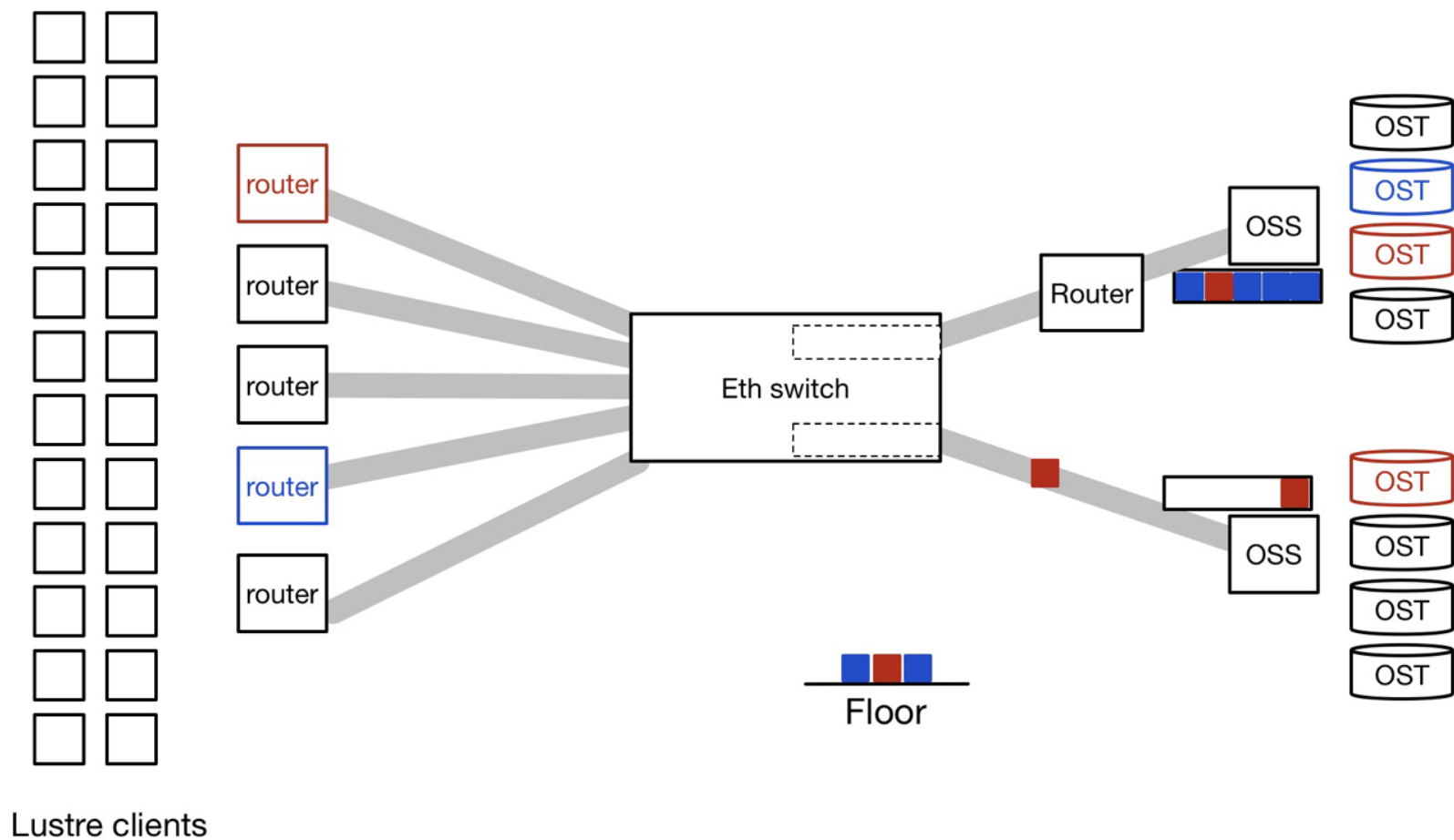
Lustre clients

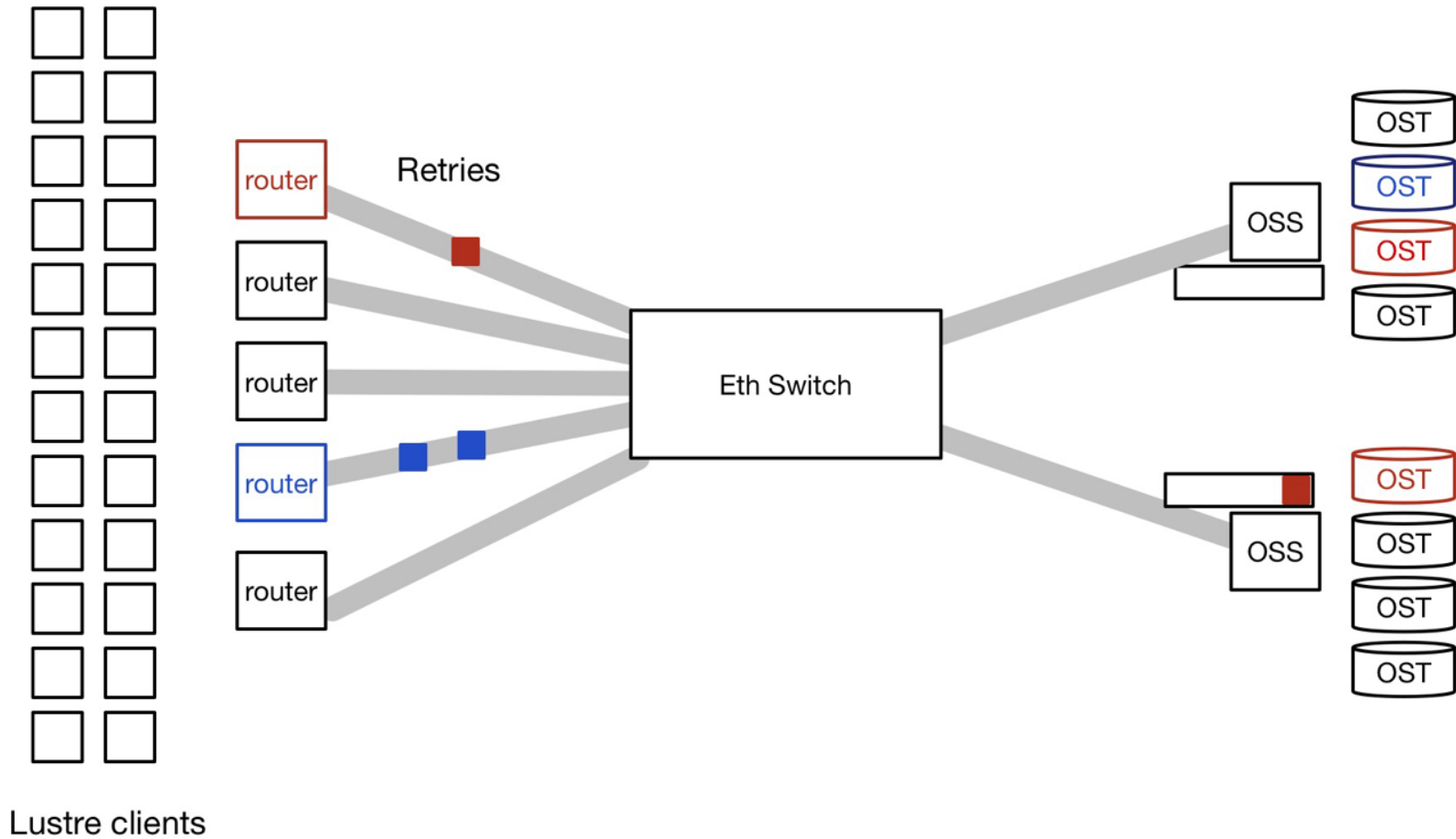




Lustre clients







Performance under Congestion

Infiniband

- Credit based flow-control will hold up messages, but they will be buffered without drops
 - Near full utilization on-the-wire
 - **Immediately resume transmission at full rate**
- Up to 15 VLs with separate rx/tx buffers

Ethernet

- Congestion signaled by packet drops
 - Too late: window size cut in half, dropping throughout
- All service classes compete for shared buffers
 - An overrun caused by one class will affect all others

Datacenter Network Integration

Infiniband

- Usually fabric is an island in datacenter
- Can share fabric between storage (LNet) and compute (MPI)
- Specialty tools available for diagnostics (wireshark for LNet), and monitoring
- Protocol interoperability through application layer (LNet routers) or bridging equipment

Ethernet

- Compatible with existing infrastructures (LAN/WAN)
- Converged fabric (management Eth, IPMI, LNet)
- Rich toolsets for access control, diagnostics, and monitoring
- L3 routers support varied interface types and the framing

Long-haul Network Considerations

Infiniband

- Range extenders can frame IB over other transports.
 - Obsidian Longbow turns one IB link into three to manage flow control credits

Ethernet

- Many options to bridge L2 over L3 (overlays/tunnels)
- Lustre runs over TCP, so can just be routed at L3
 - This means store/forward delay at every hop
- Requires large buffers (bandwidth-delay product)

Tuning Complexity

Infiniband

- Fabric-wide routing and QoS configuration done on subnet manager
 - More of a plug and play experience for small fabrics

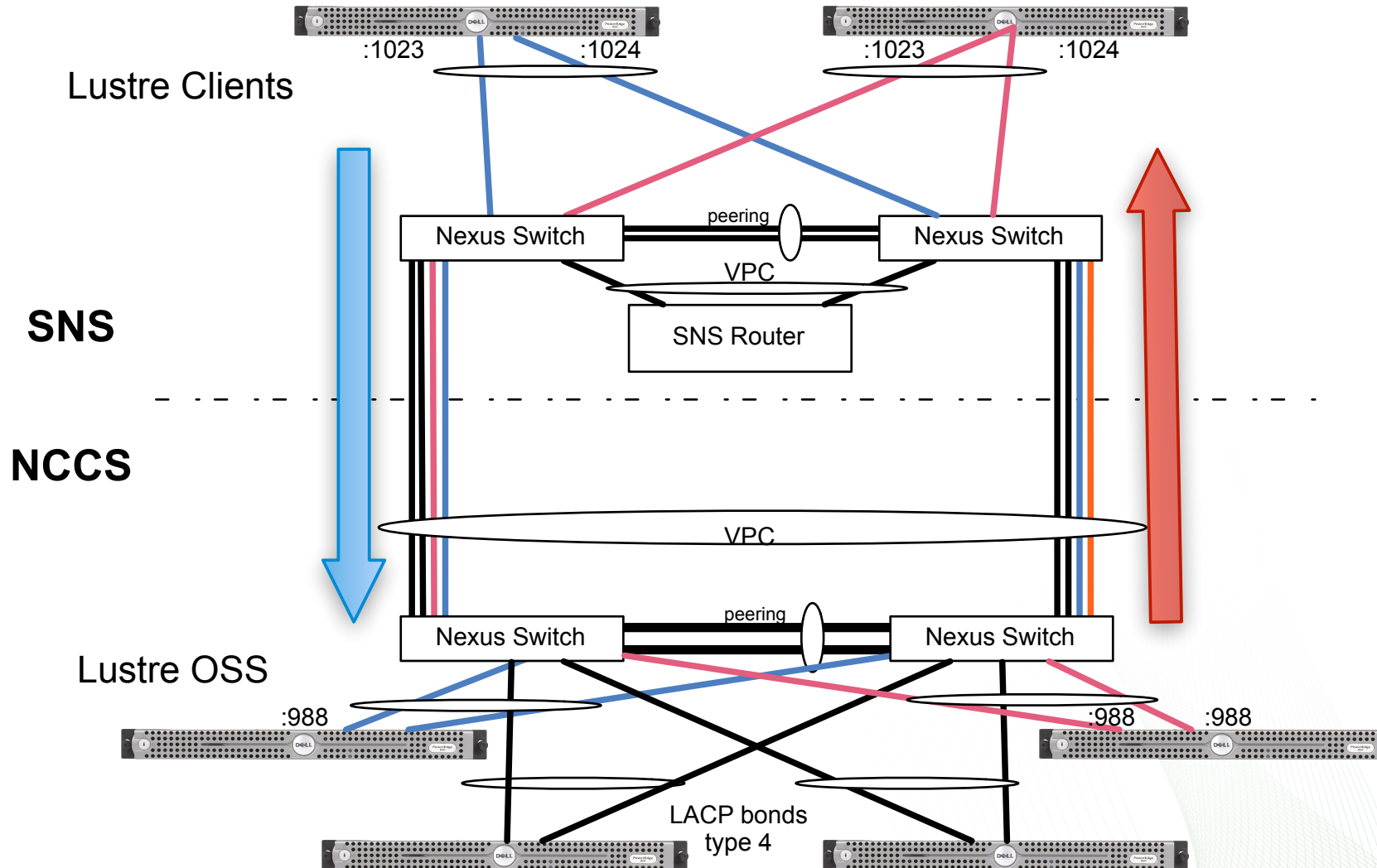
Ethernet

- E-E performance requires matching settings (flow control, MTU) on every link
 - Difficult to get consistent performance

Case Study

- A Lustre deployment for Spallation Neutron Source at Oak Ridge National Laboratory
- 448TB, 4OSS/1MDS, Lustre 1.8, 2x10GE (channel-bonded), DDN SFA10K.
 - Backend is capable of 12GB/s (verified with xdd)
 - LNET capable of 8GB/s
- 1-2miles of fiber between SNS and NCCS (ORNL)

SNS LNet design: redundancy through LACP Bonds



Application Results with 2x10GE

- Single client FS write ~ **2.1 GB/s (16.8 Gbit/s)**
 - 6 threads (single-thread limited to ~900MB/s)
 - Separate files for each thread (lock contention)
- Parallel file copy ~ **1.58 GB/s (12.6 Gbit/s)**
 - NASA's mcp, cache to disk file copy
 - Direct I/O, double-buffering, 4 threads
- How fast can dd go? ~ **900 MB/s (7 Gbit/s)**
 - Single LNET connection means no hashing
 - Lustre osc checksums off
- 32 node IOR ~ **2-4 GB/s (10GB/s with IB)**

Summary/Recommendations

socklnd vs. o2iblnd

- o2iblnd for low-latency **consistent** performance
- socklnd can compete with o2iblnd in terms of bandwidth **when parallelism is low**
- socklnd is best for heterogeneous clients
 - Facility-wide filesystems
 - Cloud use cases
- Use both!
 - Multi-homed LNET

Resources

- “Ethernet v. Infiniband”
 - <http://www.informatix-sol.com/docs/EthernetvInfiniBand.pdf>
- Jason Hill – “Lustre Tuning and Advanced LNET Configuration”
 - <http://lustre.ornl.gov/lustre101-courses/content/C1/L5/LustreTuning.pdf>
- Chris Horn – “LNET and LND Tuning Explained”
 - http://www.eofs.eu/fileadmin/lad2015/slides/15_Chris_Horn_LAD_2015_LNET.pdf
- Doug Oucharek – “Taming LNET”
 - http://downloads.openfabrics.org/Media/IBUG_2014/Thursday/PDF/06_LNet.pdf

Questions, please

blakec@ornl.gov

Case study backup slides



Network Validation

- Look for ~90% actual throughput (e.g. 9Gb/s out of 10GE) – iperf/netperf
- Look for packet loss at 9Gb/s with UDP
 - `iperf -w8m -u -l 16384 -c 10.x.x.x -b9G -i 2`
- Verify 9K MTU clean path
 - `ping -s 8972 -Mdo 10.x.x.x`
- Channel bonding complicates troubleshooting individual links (have to systematically “break” the bonds)

Latency Measurement

- NetPIPE measurements (8192 byte messages)
 - 105 μ s between sites (1 mile)
 - Not representative of WANs
 - 75 μ s on same switch
 - So a 30 μ s delay from fiber path and L3 hops
 - For comparison: 40 μ s host-to-host (no switch), 20 μ s IPoIB HCA-to-HCA

NIC Tuning

- Set IRQ affinity according to NUMA topology
- Interrupt coalescing set according to workload
- Turn on TCP SACK on (`net.ipv4.tcp_sack`)
 - Old Mellanox IB tuning script turned off, but OSS had both IB and Ethernet interfaces
 - Symptom was conflicting iperf tests sometimes 9Gb/s, then 1Gb/s. Repeatable, but independent of direction.

Host Kernel and PCI Tuning

- Sysctl parameters (<http://fasterdata.es.net>)

```
# receive window
net.ipv4.tcp_no_metrics_save = 0
net.ipv4.tcp_window_scaling = 1
# congestion control
net.ipv4.tcp_congestion_control = htcp
net.ipv4.tcp_timestamps = 0
# for ethernet networks
net.ipv4.tcp_sack = 1
```

Keep congestion window large

cubic is another good option

Accommodate packet loss and reordering

- Verify PCI capabilities

```
# lspci -vv
MaxPayload 128 bytes, MaxReadReq 4096 bytes
```

Viewing TCP Stats from Lustre

- *lctl conn_list*
 - List active TCP connections, type (I=bulk in, O=bulk out, C=control)
 - Note tx_buffer_size/rx_buffer_size determined by TCP auto-tuning in kernel
- **Example: sns-client writes to sns-oss4**

```
[root@sns-client ~]# lctl --net tcp conn_list
12345-128.219.249.38@tcp O[14]sns-client.ornl.gov->sns-oss4.ornl.gov:988 5863480/87380
nonagle
12345-128.219.249.38@tcp I[13]sns-client.ornl.gov->sns-oss4.ornl.gov:988 65536/87380
nonagle
12345-128.219.249.38@tcp C[9]sns-client.ornl.gov->sns-oss4.ornl.gov:988 65536/3350232
nonagle
```

```
[root@sns-oss4 ~]# lctl --net tcp conn_list|grep sns-client
12345-128.219.249.34@tcp I[2]sns-oss4.ornl.gov->sns-client.ornl.gov:1021 65536/16777216
nonagle
12345-128.219.249.34@tcp O[1]sns-oss4.ornl.gov->sns-client.ornl.gov:1022 65536/87380
nonagle
12345-128.219.249.34@tcp C[0]sns-oss4.ornl.gov->sns-client.ornl.gov:1023 65536/1492168
nonagle
```

Max

Observing Effect of Credits

- Flow-control by peer_credits

- ksocklnd module options on server (128.219.249.34): credits=4
peer_credits=2
- lst with --concurrency 3 (more than peer_credits, less than credits)

```
/proc/sys/lnet/nis:
```

nid	status	alive	refs	peer	rtr	max	tx	min
128.219.249.34@tcp	up	-1	1	2	0	4	4	4

Reflects LND
parameters

```
/proc/sys/lnet/peers:
```



nid	refs	state	max	rtr	min	tx	min	queue
128.219.249.45@tcp	4	up	2	2	2	-1	-2	3145944

peer_credits exceeded, so
there is tx queuing (negative
credits).

“High water mark” is -2.

Lustre Parameters

- `osc.*.checksums`
 - Without checksums: single-threaded writes up to 900MB/s
 - Still have TCP checksums
 - With checksums: 400-600MB/s
- `osc.*.max_rpcs_in_flight`
 - Increase for small IO or long fast network paths (high BDP)
 - Decreasing imposes flow-control before TCP congestion control
 - Increase to fill pipe if bandwidth-delay product is high

Bandwidth-delay product		$2 \times \text{BW (10Gb/s)} \times \text{Latency (105}\mu\text{s)}$
		$275 \text{ kB} < \text{max_rpcs_in_flight} \times \text{RPC size}$

LNet Self-test Commands

- `lst add_test --concurrency [~max_rpcs_in_flight]`
- `lst add_test --distribute 1:1`
 - Expect 1150 MB/s out of each pair with concurrency
- `lst add_test --distribute 1:4 --concurrency 8`
 - Look for improvements from hashing across bonds
- `lst add_test --distribute 4:1 --concurrency 8`
 - Evaluate congestion control settings
- Take packet header capture with `tcpdump`
 - Verify congestion window sizing
 - Bandwidth efficiency – % of throughput lost to TCP packet loss and congestion window ramping

Running LNet Self-test

- **Single stream baseline: 698MB/s**

lst add_test --batch bw_test --loop 8192 --concurrency 1 --distribute 1:1 --from c --to s brw read size=1M

```
/proc/sys/lnet/peers:
```

nid	refs	state	max	rtr	min	tx	min queue
128.219.249.45@tcp	2	up	8	8	8	7	6 1048648

```
[LNet Rates of s]
```

```
[W] Avg: 1397      RPC/s Min: 1397      RPC/s Max: 1397      RPC/s
```

```
[LNet Bandwidth of s]
```

```
[W] Avg: 698.37   MB/s  Min: 698.37   MB/s  Max: 698.37   MB/s
```

- **Setting concurrency to 16 maxes out 10GE (no hashing for 20GE)**

```
/proc/sys/lnet/peers:
```

nid	refs	state	max	rtr	min	tx	min queue
128.219.249.45@tcp	15	up	8	8	8	-6	-9 11535824

```
LNet Rates of s]
```

```
[W] Avg: 2363      RPC/s Min: 2363      RPC/s Max: 2363      RPC/s
```

```
[LNet Bandwidth of s]
```

```
[W] Avg: 1181.56  MB/s  Min: 1181.56  MB/s  Max: 1181.56  MB/s
```